Sinav Soruları

12.11.2019

1

```
int x=10;
Static int j=4, y=6;
Void printfxy(int);
int main(){

    int x;
    for(x=0; x<2; x++){
        int y=3, k=0;
        printf("X1=%d y1=%d j1=%d",x,y,j); // 0,3,4 ; 1,13,3,0
        printfxy(k); //10,13,3 ; 10,14,0
    }
    if(y==6) printf("esittir"); //esittir

    return 0;
}
```

```
Void printfxy(int a){
    Static int y=12;
    y++;
    printf("%d %d %d",x,y
        --j,a++);
```

2

```
char* x[5]={"Dandanakan1040","Mercibadık1516", "Preveze1538", "Trablusgarb1911",
            "Kurtulus1919"};
char* y[]={x[0], x[3]};
char* p=y[0];
printf("%s", y[1]);  ——→ Trablusgarp1911
printf("%s", (p+1));  ——→ andanakan1040
printf("%c", *(*(x+2)+3));  ——→ v
printf("%c", *(*(y+1))[4]);  ——→ hatalı
printf("%c", *(++y[1]));  ——→ r
printf("%c", x[1][9]+x[1][11]-x[1][13]);  ——→ j
```

③

```
int j=4, k=4.5, m=-8;
float x=0.5, y=6, z;
char a='1', b='5', c=1, d;
```

d=(a==c) ⟶ d=0

%c, b-c ⟶ 4

%d, b-c ⟶ 4

z=-3/y+k ⟶ 3.5

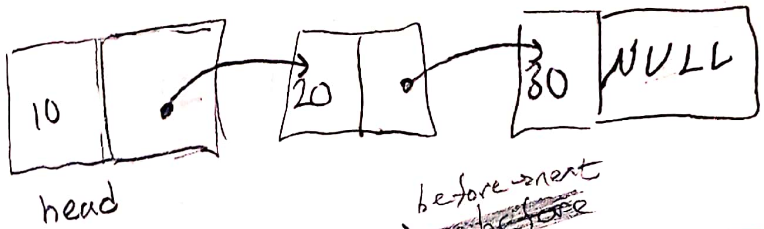
z= k+m/x ⟶ -6

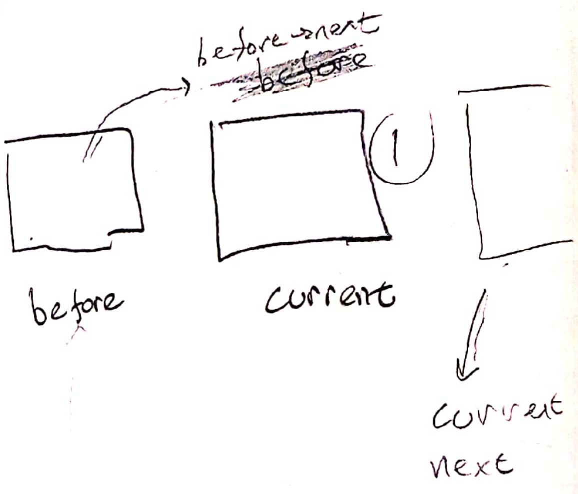
(j++)-(--m) ⟶ 4 - -6 = 10

④

```
int adet=0, i, run, length, j;
int* temp;
for(i=0; i<n/2; i++) {
   run= dizi[i*2];
   length= dizi[i*2+1];
   temp= (int*)realloc (new dizi, (length+adet) *sizeof(int));
   new dizi = temp;
   for( j=0; j<length; j++) {
      new dizi [j+adet]= run;
```

```
Struct node {
        int val;
        Struct node* next;
};
```





```
int main() {
    struct node* head, *newhead, *dummy;
    int num;
    head=(struct node*) malloc (sizeof (struct node));
    scanf("%d", &num); //10
    head->val = num;
    head->next = NULL;
    scanf("%d", &num); // 20
    push(head, num);
    scanf("%d", &num); //30
    push(head, num);
    list(head);
    scanf("%d", &num);
    newhead = pushFront (head, num);
    list(newhead);
    scanf("%d", &num);
    if (num==newhead->val) {
       dummy = newhead->next
       free(newhead);
       newhead = dummy;
    }else
    deleteNode(newhead, val);
}
```

```
Void push(struct node* head, int var) {
    Struct node* current= head;
    While (current->next != NULL) {
        current = current->next;
    }
    Current->next = (struct node*) malloc (sizeof (struct node))
    Current->next->val = var;
    current->next->next = NULL;
}
Void list (struct node* head) {
    struct node* current = head;
    While(current->next != NULL) {
        printf("%d", current->val);
        Current = current->next;
    }
    printf("%d", current->val);
}
```

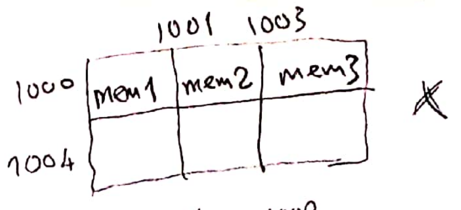
```
Struct node*  pushFront (struct node* head, int val){

        Struct node* newN;
       newN = (struct node*)malloc (sizeof (struct node));
       newN → next = head;
      newN → val= val;
      return newN;
}
```

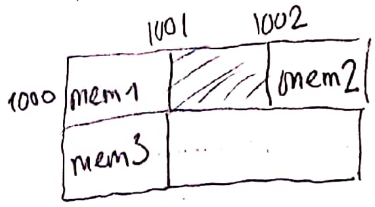
(2)

```
Void deleteNode (struct node* head, int val){
    Struct node* current = head, *before = NULL;
    While ((current → val != val) && (current → neet != NULL)) {
        before= current;
        Current = Current → next;
    }
    if (current → val == val ) {
       printf("NULL");
    }
    else{
        before → next = Current → next;
        free (current);
    }
}
```

Alignment of Structure Members

```
Structure ALIGN {
        Char  mem1;
        Short mem2;
        char  mem3;
    } s1;
```

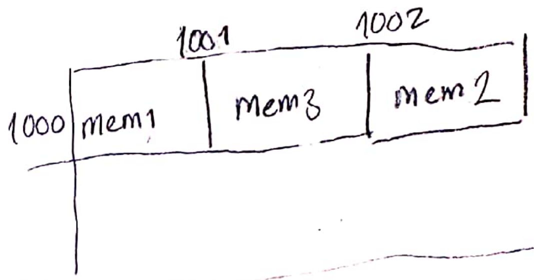




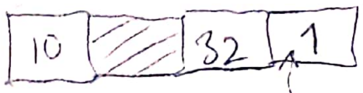
↓

```
Structure ALIGN {
        Char mem1,mem3;
        Short mem2;
    }s1;
```

Naturally Aligned



```
typedef struct {
        char a;          Char a,c;
        short b;         short b;
        char c;
    }ABC;

int main() {
    ABC v;
    Char* pv;
    V.a=10;
    V.b=32;
    V.c=1;
    pv= &(p.a);
    printf("%d",*pv); //10
    pv=pv+sizeof(char)+sizeof(short);
```



Bitfields

basetype

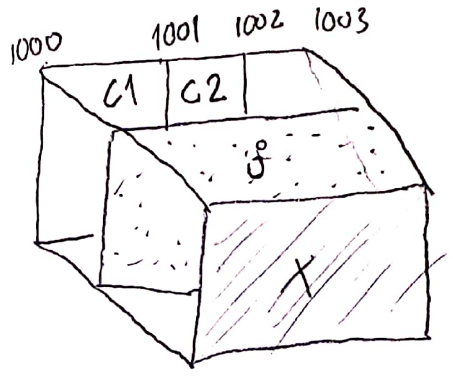
unsigned int
signed int
short
char

bitfield name $\longrightarrow$: bitlength;

```
unsignet int day:5;
    int too_long:17;  ——→ 16 bitlik sistem için illegal

typedef struct{
        unsigned int day:5;
        unsigned int month:4;
        unsigned int year:11;
      }DATE;
int main(){

    DATE dizi[3];     int tarih1,tarih2;
    scanf("%d", &dizi[0].day);  ——→ illegal X
    scanf("%d", &tarih1);  }
    dizi[0].day= tarih1;   }  ——→ legal ✓
```

UNIONS

```
typedef union {
        struct {
                char c1,c2;
        } s;
        long j;
        float x;
    } U;
```



```
U example;
  example.s.c1='a';
  example.s.c2= 'b';
  example.j = 5;      ──→ c1'de 5'in ilk byte'ı var
```

```
union doub{
        unsigned char c[2];
        unsigned short val;
    }

union doub d;
d.c[0]=1; →ØØØØØØØ1
d.c[1]= 1; → ØØØØØØØ1
printf("%d",d.val); → ØØØØØØØ1ØØØØØØØ1 → 257
```

Initializing Unions

```
union u {
        struct {
                char f1, f2;    → 1 2
                short f3;       → 3
                } s;
        unsigned char f4[6];  → 1 2 3 0 0 0
};

int main() {
    union u test = {1, 2, 3, 4, 5, 6};
    printf("%d %d %d", test.s.f1, test.s.f2, test.s.f3); → 1/2/3 ↰
    union u test = {.f4 = {1, 2, 3, 4, 5, 6}};
```

C ~~PREPRO~~PROCESSOR

#define — macro processing

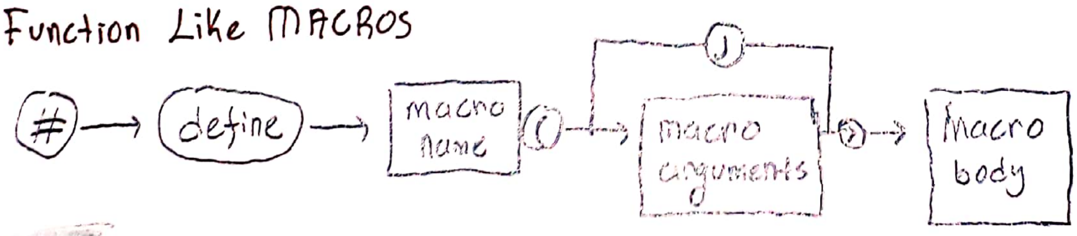
#include — inclusion at additional file (#include)

— conditional compilation (#if, #elif, #else, #endif)

```
#define LONG_MACRO "This is very\
     long macro"
#define N 100          → Gelenek / okunabilirlik için büyük harf
#define BIG_BUFF 512

int main() {
   char kelime[BIG_BUFF]; // char kelime[512];
   double numbers[N]; // double numbers[100];
   return 0;
}
```

MACROS

Function Like MACROS



```
#define MUL_BY_TWO(a) ((a)+(a))
                        └──→ parantez içine alınmalı
j= MUL_BY_TWO(5);
        └──→ j= ((5)+(5));
```

```
#define square(a) a*a  ──→ square(a) ((a)*(a)) olması istediği gibi olur.

J= 2*square(3+4) // 2* 3+4 * 3+4
```

```
#define min(a,b) ((a)<(b)?(a):(b))
a= min(3,5); // a= ((3)<(5)?(3):(5));
x=1, y=5;
z= min(++x, y);
z= ((++x)<(y)?(++x):(y)) // z=3 olur
```

```
#define MUL-BY-TWO(a) ((a)+(a))

int y, z;

y= MUL-BY-TWO(2.4); // 4

z = MUL-BY-TWO(2.4); // 4.8

#undef MUL-BY-TWO
```

Built in Macros

```
--LINE--     satir no          printf("this program compiled on %s at %S",--DATE--
                                                                        ,--TIME--);
--FILE--
--TIME--
--DATE--
--STOC--
```

```
#define CHECK(a,b)\
        if((a) != (b)) \
        fail(a,b,--FILE--,--LINE--)

Void fail (int a, int b, char* p, int line) {
     printf("check failed in file %s at line %d: received %d expected %d",
                                                                p, line, a, b);
}
int main () {
    CHECK(3,4);
    return 0;
}
```

LINUX

Command Line Arguments

hesap.c

```
./hesap topla 4 6
```

```
#include<stdio.h>
#include<string.h>

int main(int argc, char* argv[]){
   int sayi1,sayi2;
   if(argc!=4){
     printf("yanlis");
     exit 0;
   }
   sayi1= atoi(argv[2]); // converts string to integer
   sayi2= atoi(argv[3]);
   if(!strcmp("topla", argv[1]){
       printf("%d", sayi1+sayi2);
   else{
       printf("%d", sayi1-sayi2);
   }
   return 0;
}
```

```c
#include<stdio.h>
int main(int argc, char* argv[]){

      while(--argc>0){
            printf("%d",argc);
            printf("%d\n",*++argv);
      }
      return 0;
}
```

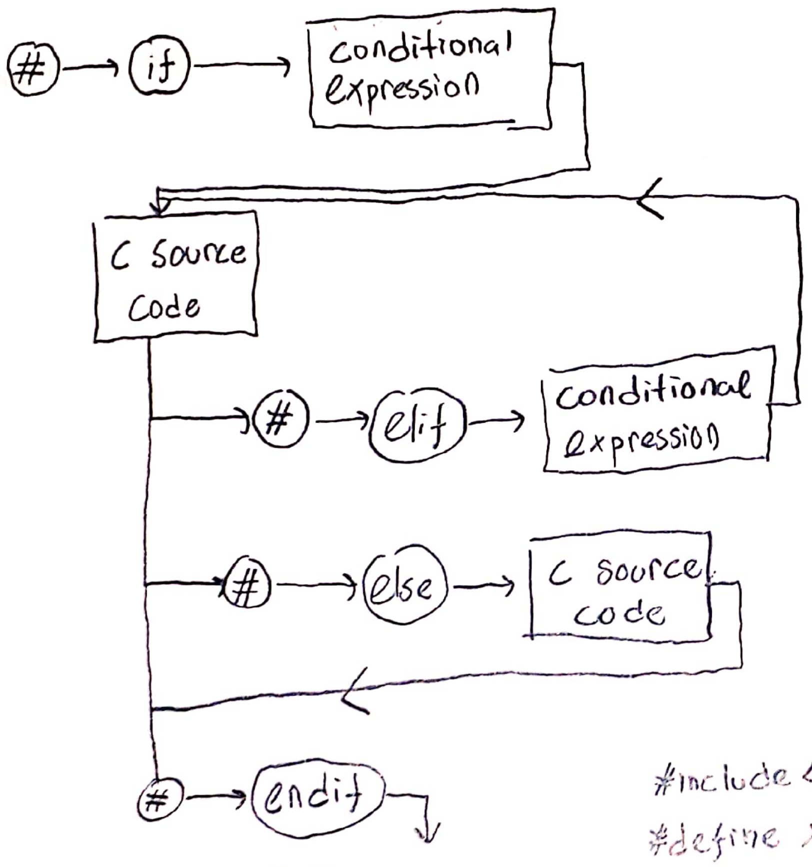
```
./yaz irem 1 2 3
```

```
4 irem
3 1
2 2
1 3
```

output

Conditional Compilation

#if #else #elif #endif



```
#if x==1
    #undef x
    #define x Ø
#elif x==2
    #undef x
    #define x 3
#else
    #define y 4
#endif
```

```
#include <stdio.h>
#define x 2
int main(){
  #if x==1
    printf("oyle");
  #elif x==2
    printf("boyle");
  #endif
return 0;
}
```

Testing Macro Existence

#ifdef #ifndef #endif

#if defined X ≈ #ifdef X

#if !defined X ≈ #ifndef X

```
#ifdef TEST
    printf("this is a test");

#else
    printf("this is not a test");
#endif
```

~

#if X // 0 döndürür.

~

```
#ifndef FALSE
  #define FALSE Ø

#elif FALSE
  #undef FALSE
  #define FALSE Ø
#endif
```